STAR GATE

~~SECRET~~

DT-S-1036-SL

DEFENSE
INTELLIGENCE
AGENCY

PROFICIENCY ENHANCEMENT PROJECT (U)

PHASE I

6 JUNE 1991

NOFORN
~~SECRET~~
LIMDIS

STAR GATE

~~SECRET~~

PROFICIENCY ENHANCEMENT PROJECT
PHASE I

Date of Publication
6 June 1991

This is a Department of Defense Intelligence Document prepared by the Technology Assessment and Support Office, Directorate for Scientific and Technical Intelligence Defense Intelligence Agency.

PREPARED BY

Technology Assessment and Support Office (DT-S)

NOT RELEASABLE TO FOREIGN NATIONALS

LIMITED DISSEMINATION

REPRODUCTION REQUIRES
APPROVAL OF ORIGINATOR
OR HIGHER DOD AUTHORITY

FURTHER DISSEMINATION
ONLY AS DIRECTED BY DT
OR HIGHER DOD AUTHORITY

CLASSIFIED BY: DIA/DT
DECLASSIFY ON: OADR

~~SECRET~~/NOFORN/LIMDIS

STAR GATE

TABLE OF CONTENTS

PROFICIENCY ENHANCEMENT PROJECT
PHASE I

I. (U) PURPOSE:

(S/NF) This paper documents the completion of the first phase of a DT-S originated training/applications research project.

II. (U) SCOPE:

(U) This report discusses the concept, documents the unique hardware and software approach developed, and identifies instructions/protocols involved in this proficiency enhancement activity.

(S/NF) Follow-on reports of this proficiency enhancement project are anticipated that will discuss later phases. These reports will address statistical results, evaluation criteria, training/performance rates, and potential impact on select applications/projects.

III. (U) DISCUSSION:

(S/SG/NF/LIMDIS) This proficiency enhancement project was developed by DT-S in order to improve the efficiency and pace of training/practice available to DT-S personnel. It is anticipated that this technique will significantly enhance the success probability of certain applications of interest, such as search/track and possibly other operational projects. This training/practice technique could also lead to the possibility of a long-distance "communications" capability. Even though data-rate would probably be limited, there may be some situations where such a capability would be useful (e.g., hostage projects).

IV. (U) BACKGROUND:

a. (S/SG/NF/LIMDIS) Several members of the DT-S staff attended the 1991 Atlantic University Consciousness Symposium, held in Virginia Beach, VA, on 16-17 February 1991 (see report DT-S-1029-SL, dtd 15 MARCH 1991). While there, two members of the staff were allowed to use an antique Zener card machine in the university's possession, which purportedly tested "mental communication" skills (a diagram of this machine is shown in Appendix A). Results were quite successful, prompting later discussion about the experience. The idea surfaced that this technique would also be of direct use to DT-S as a potential proficiency enhancement procedure.

b. (S/SG/NF/LIMDIS) Work toward the goal of developing such a capability began upon return from the symposium and a computerized version of this antique "mental communication" testing device was developed. Modifications would be needed, however, to permit this technique to have a wider role for DT-S. Thus, a variety of training/practice approaches would be incorporated, the system would be designed for very rapid use (including feedback), results would be part of a permanent record for each individual, and rapid statistical evaluations could be made.

V. (U) DESIGN CRITERIA FOR HARDWARE AND SOFTWARE:

(S/NF) Even though a variety of training/practice procedures could be accomplished from the envisioned proficiency training technique, the most stringent aspect from a designer point-of-view was the "communications" aspect. This, as well as other needs led to the following requirements:

a. (S) Accessibility for all DT-S staff.

b. (U) Targets selected at random from a large target pool.

c. (U) Statistical track of individual and group performance maintained to include:

1. "Receiver's" name or Source number.
2. "Sender's" name or Source number.
3. Date of session.
4. Number of trails.
5. Number of successful trials.

d. (U) Format appropriate for periodic status and progress reports.

e. (U) Flexibility: This system requires flexibility in order to allow scientific experimentation with different types of targeting groups. Zener cards only provide a set of geometric and meaningless symbols; this system should include human expressions, colors, emotional situations, etc.

f. (U) Built-in potential for unlimited target pools to permit expansion of application options.

g. (U) Communications utility: In this case, the system would require additional software to permit utility of coded messages to be evaluated. This additional software can be generated in-house.

V. (U) IMPLEMENTATION: A study indicated that the majority of equipment/technology already existed in DT-S. Therefore, the following guidelines were considered for project implementation:

a. (U) Existing computers were used. This further meant that there would be no terminals dedicated to this project alone, but that all hardware and software must be "invisible" to normal operations.

b. (U) The two computers had to be "tied together" so they could communicate. This was necessary in order to allow both to agree on the same target card, and so both would be aware of the "receiver's" card choice.[1]

d. (U) DT-S uses the TURBO PASCAL programming language, which is completely adequate for the programming demands for this project. SFC SG1J [redacted] of DT-S is a qualified programmer with this language, and used it to write the three programs necessary for the project. These three programs are:

1. MENTCOMM.EXE This is the "master" program which perform all required functions for the user, selects targets, controls both machines, and performs all necessary data manipulation functions. The programming source code

[1] (U) Computers can not be internally modified without permission from DSO. A request would have to delineate all aspects of the project, requiring DSO people to be read on, etc. Therefore, all connecting and control circuitry had to be external and "invisible" to normal operations. A full study of the possibilities finally led to the simplest solution, a "gimmicked" inter-computer control cable (acting as a nine-wire "null modem"), as depicted in Appendix A.

listing for this program is located in Appendix B.

2. MENTCOM2.EXE This is the "slave" program which is run on the "sender's" machine. This program only serves to display the target card to the "sender". The programming source code for this program is located in Appendix C.

3. PI3DECOD.EXE This program is used by the other two, and not by the user, directly. The programming sources code for this program is located in Appendix D.

e. (U) Along with these three programs, other files are used to contain the libraries of target photos, diagrams, symbols, etc. A hard copy of the present target file libraries are found in Appendix E. Each library holds 18 targets, and the program is expandable to use as many target libraries as future needs warrant.

VI. (U) OTHER CONSIDERATIONS:

a. (C) Because software is so easily transportable, and often seems to appear in unauthorized places, as if by magic, the title page of this program was written without any reference to DIA, DT-S, Project STAR GATE, or the U.S. government. For all outward appearances, it is a general purpose program, of the type which could be bought in a store. Likewise, no classified material has been or will be put into the target libraries.

b. (C) Since practice time is rare, it was believed that opportunities for two people to work together in practice would be even rarer. For this reason, the program was so designed to automatically know whether or not it was being used by one person or two. If a person wants to practice alone, the "master" program will continue to operate, but the emphasis will be on "communicating" with the computer, rather than with another person. The data which is kept at the end of the practice session will reflect the fact that a solo session was performed.

c. (S) Since there is strong evidence from other experiments indicating that the type of target affects a person's ability to mentally communicate, the libraries are oriented toward "sets" or target types. Data records kept at the end of each session also indicate which type of target library was used, allowing for future analysis of each source's performance according to target type.

VII. (U) INSTRUCTIONS FOR PERFORMING A SESSION: Every attempt has been made to make the program as "user-friendly" as possible.[2] Operation is as follows:

a. (U) If two computers are to be used in tandem (for the "sender/receiver" mode of operation), the inter-computer connecting cable must be plugged into the COM2 port of each machine. As soon as we can establish a dedicated location and permanent hook-up for this project, this step will no longer be required.

b. (U) The computer(s) is(are) turned on, and when the main menu appears, the option for Mental Communications Practice is selected.

c. (U) A second menu appears, asking whether the user will be acting as "sender" or "receiver".

1. If the user is the one who will try to acquire the targets, he/she is the "receiver" (of the mental communication), and should select the (R)eceiver option.

2. If the user is the one who will be looking at the target picture and trying to "send" this information to the other, he/she should select the (S)ender option.

d. (U) The instructions for each person will appear on their individual screens. From this time on, the person acting as "receiver" is in complete control of the programs on both machines.

[2] (U) All computer screens seen during the program are shown in hard copy form in Appendix F.

e. (U) When both participants have read the instructions, the "receiver" person presses a key, and is given the choice of selecting which library of pictures/diagrams/symbols/etc. will be used for the session.

f. (U) Once the target library has been selected, the complete library will be shown on both screens. The reason for this is to prevent overdue attraction to any one target picture during the later practice session, simply because it has not been seen for a while.

g. (U) When the library of targets has been removed from the screen, the "receiver's" (main) program will randomly select four target pictures from the library and display them side-by-side on the "receiver's" screen. The computer will then randomly select one of these four as the specific target. A copy of the specific target will be shown on the "sender's" screen (on the other computer).

h. (U) The "sender" concentrates on the target picture, imagining him/herself interacting with the picture's contents, studying the specific aspects of the picture, etc.

i. (U) The "receiver" waits for an impression. This may be visual, auditory, or just a "feeling" for one of the four pictures over the others. Each picture has a number underneath it (1..4). When the "receiver" has determined which picture to select, he/she presses the 1, 2, 3, or 4 key on the computer keyboard.

j. (U) IMMEDIATE FEEDBACK IS THEN PROVIDED. The screen will blank for a moment, then the numbers will reappear. However, only the target picture will reappear above its number. This image is held on the screen for approximately one second, before moving on to the next trial.

k. (U) The number of trials completed is shown in the lower left hand corner of the screen, and the number of "hits" is shown in the lower right hand side of the screen during each trial. The number of "hits" is also shown to the "sender" on the other computer, as positive feedback to him/her.

l. (U) When the "receiver" is ready to quit the session, the Esc key is pressed. The screen will blank, and a complete statistical package will be shown on both computer screens. At this time, the "receiver" has three choices:

a) To continue the session, taking up where he/she left off.

b) To start the session over again from zero.

c) To quit.

m. (U) Should the "receiver" choose to quit, the computer will ask for the date and the "receiver's" identification. If this has not been a solo session, the computer will also ask for the "sender's" identification.

n. (U) All session statistics are added to the "receiver's" data file before the program terminates, and the user is returned to the computer's menu screen.

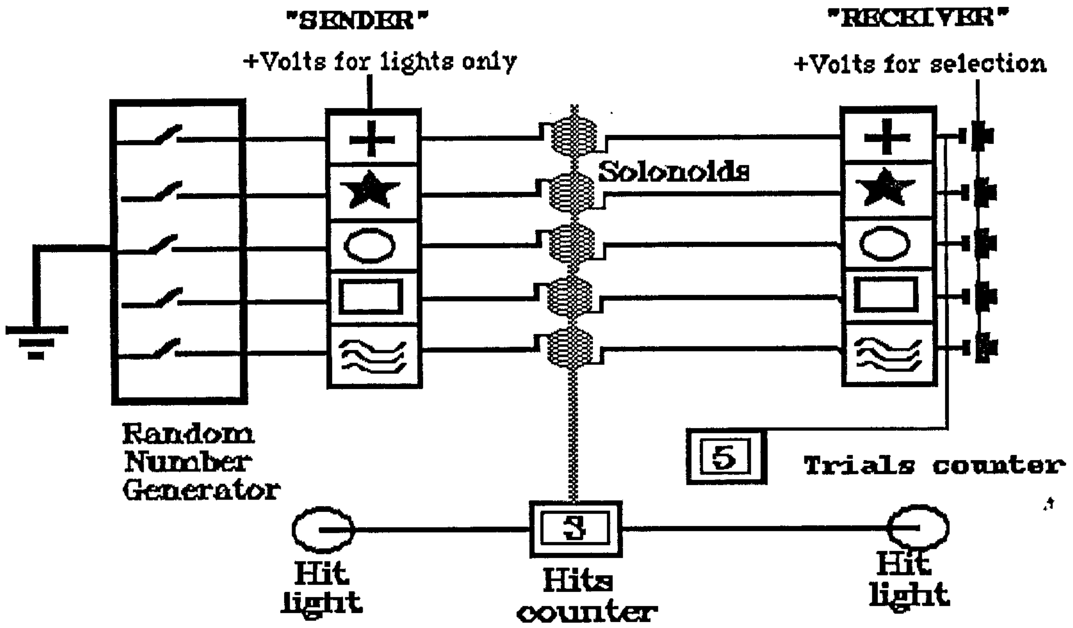


Figure A-1: ESP TESTING DEVICE AT AMERICAN UNIVERSITY
Functional Diagram

1. The random number generator randomly closes one of the five switches, lighting the corresponding light behind one of the "Sender's" symbols.

2. The "Sender" concentrates on the lit symbol.

3. The "Receiver" waits until an impression is gained and then presses the button below the symbol he/she thinks is the proper target.

4. Pressing the button automatically kicks up the Trials counter.

5. If the correct symbol is selected, the closed circuit also kicks up the Hits counter and lights the "HIT" lights as positive feedback to both participants.

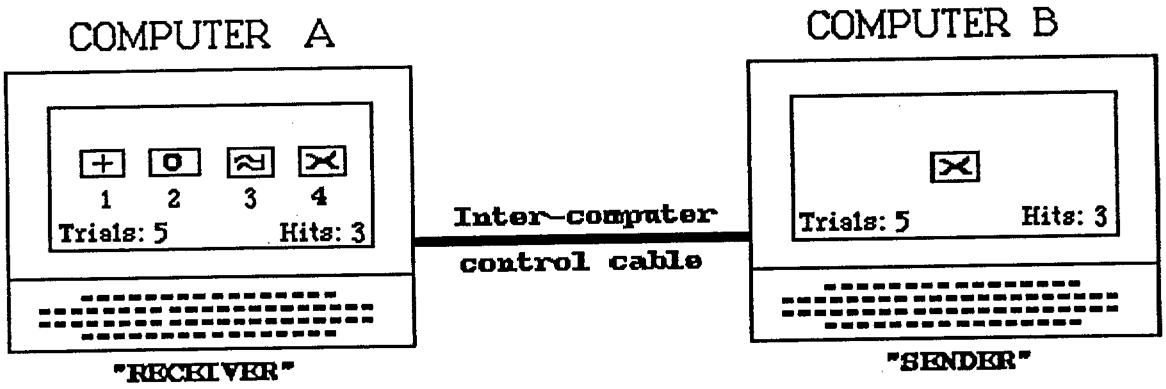


Figure A-2: ESP MACHINE FOR DT-S

1. The "Receiver's" computer selects four targets from a library of many, and shows them on the screen of Computer A.

2. It then randomly selects one of the four and sends that target to the "sender's" computer.

3. The "Sender" concentrates on the target picture.

4. The "Receiver" attempts to receive images, feelings, etc. about the target picture. When such a feeling is received, the "Receiver" selects the target picture from among the four shown, and presses the corresponding key on the keyboard.

5. Immediate feedback is given to each participant before continuing to the next set of pictures.

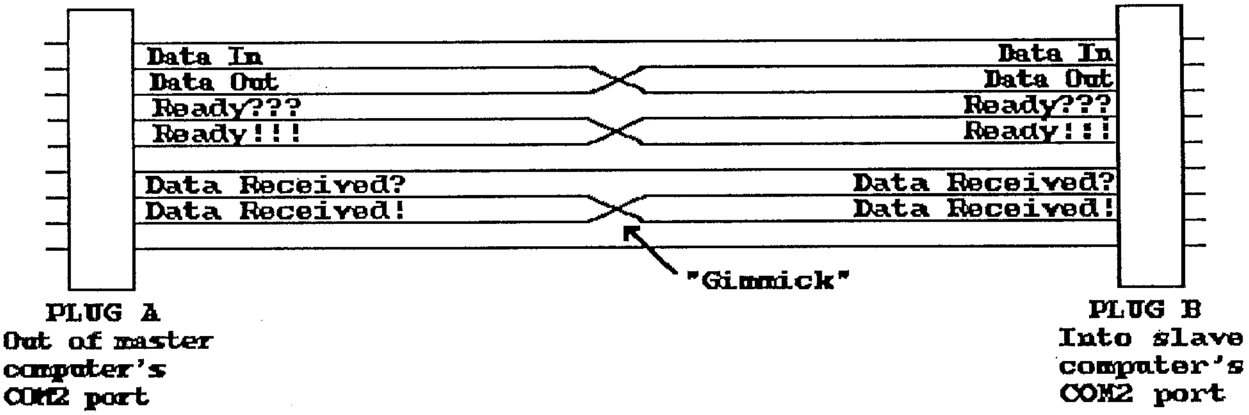


Figure A-3: INTER-COMPUTER CONTROL CABLE

Plugs are 9-pin female "D" plugs.

Cable is 9-wire flat ribbon cable.

1. Demands that no internal changes be made to the computers prompted the need for special external circuitry. This is in the form of an inter-computer control, or "Null-modem" cable, hooked to the data entry port (COM2) of each computer.

2. The cable is "gimmicked" in such a way that data out of each computer is fed into the input lines of the other, as are the request info lines ("Ready???" and "Data Received?") and the status lines ("Ready!!!" and "Data Received!")

3. During the actual use of the program, these data lines are taken over by the programming, instead of being used in the normal manner. Controlling commands are converted by MENTCOMM.EXE into the appearance of data. Once converted, they can be passed as data over the data lines. MENTCOM2.EXE reconverts the data into commands, and obeys them.

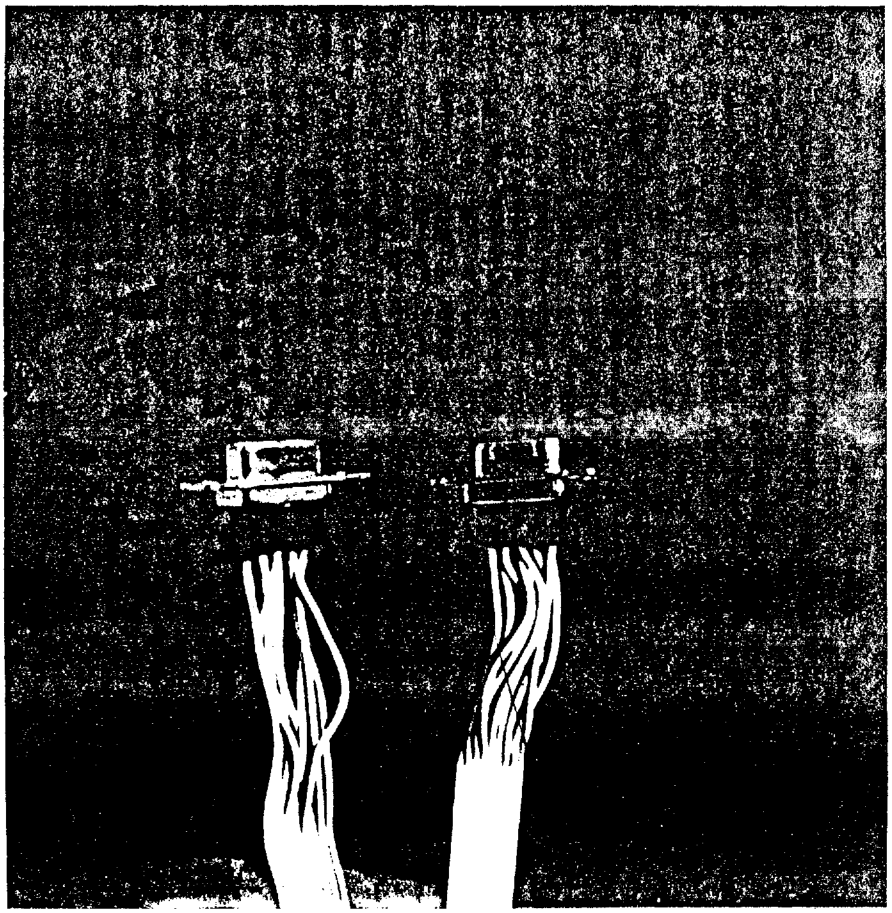


Figure A-4: Photograph of the inter-computer control cable, shoving plugs and "gimmick" areas.

SOURCE CODE FOR THE PROGRAM MENTCOMM.EXE WRITTEN IN TURBO PASCAL Ver. 5.1

```
PROGRAM MENTCOMM;
  USES
    CRT, DOS, GRAPH, PI3DECOD;

  CONST
    Height = 100;
    Width  = 100;
    Offset =  50;

  VAR
    Action    : string;
    Dir       : string;
    Key       : char;
    Hits      : integer;
    Lineout   : text;
    Name      : string;
    Numb1,
    Numb2,
    Numb3,
    Numb4,
    NumbX     : integer;
    NumbH,
    NumbS,
    NumbT     : string;
    P1, P2,
    P3, P4,
    P5, P6,
    P7, P8,
    P9, P10,
    P11,P12,
    P13,P14,
    P15,P16,
    P17,P18   : pointer;
    Picfile   : string;
    Rate      : real;
    Score     : real;
    Size      : word;
    Trial     : integer;
    X1, X2,
    X3, X4,
    X5, X6,
    Y1, Y2,
    Y3        : integer;
```

```
FUNCTION  SETX(Number : integer) : integer;
  begin
    case Number of
       1,  7, 13 : SETX :=    0;
       2,  8, 14 : SETX := Width;
       3,  9, 15 : SETX := Width * 2;
       4, 10, 16 : SETX := Width * 3;
       5, 11, 17 : SETX := Width * 4;
       6, 12, 18 : SETX := Width * 5;
    end;
  end;

FUNCTION SETY(Number : integer) : integer;
  begin
    case Number of
       1,  2,  3,  4,  5,  6 : SETY :=   0;
       7,  8,  9, 10, 11, 12 : SETY := Height;
      13, 14, 15, 16, 17, 18 : SETY := Height * 2;
    end;
  end;

PROCEDURE WRITE_LINE(tempstr:string);
  begin
    rewrite(Lineout);
    flush  (Lineout);
    writeln(Lineout,tempstr,#13);
    close  (Lineout);
    delay  (150);
  end;


PROCEDURE PICK_4;
  begin
    Randomize;
    Numb1 := random(17) + 1;
    repeat
      Numb2 := random(17) + 1;
    until Numb2 <> Numb1;
    repeat
      Numb3 := random(17) + 1;
    until (Numb3 <> Numb1) and (Numb3 <> Numb2);
    repeat
      Numb4 := random(17) + 1;
    until (Numb4 <> Numb1) and (Numb4 <> Numb2) and (Numb4 <> Numb3);
end;

PROCEDURE PICK_1;
  var
```

```
    tempint : integer;
  begin
    tempint := random(1000);
    if NumbS <> '0' then
      begin
        case tempint of
          000..250 : begin
                       NumbX := 1;
                       str(Numb1,NumbS);
                     end;
          251..500 : begin
                       NumbX := 2;
                       str(Numb2,NumbS);
                     end;
          501..750 : begin
                       NumbX := 3;
                       str(Numb3,NumbS);
                     end;
          751..999 : begin
                       NumbX := 4;
                       str(Numb4,NumbS);
                     end;
        end;
      end;
  end;

PROCEDURE SEND_INFO;
  begin
    str(Trial,NumbT);
    str(Hits ,NumbH);
    WRITE_LINE(NumbS);
    WRITE_LINE(NumbT);
    WRITE_LINE(NumbH);
  end;

PROCEDURE BREAK_PICTURE_UP;
  var
    x : integer;
    y : integer;
  begin
    Size := ImageSize(00,00,Width,Height);
    GetMem(P1, Size); X := SETX( 1); Y := SETY( 1);
    GetImage(X, Y, X + Width, Y + Height, P1^);
    GetMem(P2, Size); X := SETX( 2); Y := SETY( 2);
    GetImage(X, Y, X + Width, Y + Height, P2^);
    GetMem(P3, Size); X := SETX( 3); Y := SETY( 3);
    GetImage(X, Y, X + Width, Y + Height, P3^);
    GetMem(P4, Size); X := SETX( 4); Y := SETY( 4);
    GetImage(X, Y, X + Width, Y + Height, P4^);
    GetMem(P5, Size); X := SETX( 5); Y := SETY( 5);
    GetImage(X, Y, X + Width, Y + Height, P5^);
    GetMem(P6, Size); X := SETX( 6); Y := SETY( 6);
    GetImage(X, Y, X + Width, Y + Height, P6^);
    GetMem(P7, Size); X := SETX( 7); Y := SETY( 7);
    GetImage(X, Y, X + Width, Y + Height, P7^);
    GetMem(P8, Size); X := SETX( 8); Y := SETY( 8);
    GetImage(X, Y, X + Width, Y + Height, P8^);
    GetMem(P9, Size); X := SETX( 9); Y := SETY( 9);
    GetImage(X, Y, X + Width, Y + Height, P9^);
    GetMem(P10,Size); X := SETX(10); Y := SETY(10);
    GetImage(X, Y, X + Width, Y + Height, P10^);
    GetMem(P11,Size); X := SETX(11); Y := SETY(11);
    GetImage(X, Y, X + Width, Y + Height, P11^);
    GetMem(P12,Size); X := SETX(12); Y := SETY(12);
    GetImage(X, Y, X + Width, Y + Height, P12^);
    GetMem(P13,Size); X := SETX(13); Y := SETY(13);
    GetImage(X, Y, X + Width, Y + Height, P13^);
    GetMem(P14,Size); X := SETX(14); Y := SETY(14);
    GetImage(X, Y, X + Width, Y + Height, P14^);
    GetMem(P15,Size); X := SETX(15); Y := SETY(15);
    GetImage(X, Y, X + Width, Y + Height, P15^);
    GetMem(P16,Size); X := SETX(16); Y := SETY(16);
    GetImage(X, Y, X + Width, Y + Height, P16^);
    GetMem(P17,Size); X := SETX(17); Y := SETY(17);
    GetImage(X, Y, X + Width, Y + Height, P17^);
    GetMem(P18,Size); X := SETX(18); Y := SETY(18);
    GetImage(X, Y, X + Width, Y + Height, P18^);
  end;

PROCEDURE PUT_PARTS;
  var
    trialstr : string;
    hitstr : string;
  begin
    ClearDevice;
    str(Trial,trialstr);
    trialstr := 'Trial # ' + trialstr;
    SetTextStyle(defaultfont,horizdir,1);
    OutTextXY(58,315,trialstr);
    str(Hits,hitstr);
    hitstr := 'Hits: ' + hitstr;
    OutTextXY(550,315,hitstr);
    SetTextStyle(gothicfont,horizdir,3);
    case Numb1 of
      1 : PutImage(050,Height,P1^, NormalPut);
      2 : PutImage(050,Height,P2^, NormalPut);
```

```
   3 : PutImage(050,Height,P3^, NormalPut);
   4 : PutImage(050,Height,P4^, NormalPut);
   5 : PutImage(050,Height,P5^, NormalPut);
   6 : PutImage(050,Height,P6^, NormalPut);
   7 : PutImage(050,Height,P7^, NormalPut);
   8 : PutImage(050,Height,P8^, NormalPut);
   9 : PutImage(050,Height,P9^, NormalPut);
  10 : PutImage(050,Height,P10^,NormalPut);
  11 : PutImage(050,Height,P11^,NormalPut);
  12 : PutImage(050,Height,P12^,NormalPut);
  13 : PutImage(050,Height,P13^,NormalPut);
  14 : PutImage(050,Height,P14^,NormalPut);
  15 : PutImage(050,Height,P15^,NormalPut);
  16 : PutImage(050,Height,P16^,NormalPut);
  17 : PutImage(050,Height,P17^,NormalPut);
  18 : PutImage(050,Height,P18^,NormalPut);
end;
OutTextXY(050 + Offset,Height + 130,'1');
case Numb2 of
   1 : PutImage(200,Height,P1^, NormalPut);
   2 : PutImage(200,Height,P2^, NormalPut);
   3 : PutImage(200,Height,P3^, NormalPut);
   4 : PutImage(200,Height,P4^, NormalPut);
   5 : PutImage(200,Height,P5^, NormalPut);
   6 : PutImage(200,Height,P6^, NormalPut);
   7 : PutImage(200,Height,P7^, NormalPut);
   8 : PutImage(200,Height,P8^, NormalPut);
   9 : PutImage(200,Height,P9^, NormalPut);
  10 : PutImage(200,Height,P10^,NormalPut);
  11 : PutImage(200,Height,P11^,NormalPut);
  12 : PutImage(200,Height,P12^,NormalPut);
  13 : PutImage(200,Height,P13^,NormalPut);
  14 : PutImage(200,Height,P14^,NormalPut);
  15 : PutImage(200,Height,P15^,NormalPut);
  16 : PutImage(200,Height,P16^,NormalPut);
  17 : PutImage(200,Height,P17^,NormalPut);
  18 : PutImage(200,Height,P18^,NormalPut);
end;
OutTextXY(200 + Offset,Height + 130,'2');
case Numb3 of
   1 : PutImage(350,Height,P1^, NormalPut);
   2 : PutImage(350,Height,P2^, NormalPut);
   3 : PutImage(350,Height,P3^, NormalPut);
   4 : PutImage(350,Height,P4^, NormalPut);
   5 : PutImage(350,Height,P5^, NormalPut);
   6 : PutImage(350,Height,P6^, NormalPut);
   7 : PutImage(350,Height,P7^, NormalPut);
   8 : PutImage(350,Height,P8^, NormalPut);
```

```
      9 : PutImage(350,Height,P9^, NormalPut);
     10 : PutImage(350,Height,P10^,NormalPut);
     11 : PutImage(350,Height,P11^,NormalPut);
     12 : PutImage(350,Height,P12^,NormalPut);
     13 : PutImage(350,Height,P13^,NormalPut);
     14 : PutImage(350,Height,P14^,NormalPut);
     15 : PutImage(350,Height,P15^,NormalPut);
     16 : PutImage(350,Height,P16^,NormalPut);
     17 : PutImage(350,Height,P17^,NormalPut);
     18 : PutImage(350,Height,P18^,NormalPut);
    end;
    OutTextXY(350 + Offset,Height + 130,'3');
    case Numb4 of
      1 : PutImage(500,Height,P1^, NormalPut);
      2 : PutImage(500,Height,P2^, NormalPut);
      3 : PutImage(500,Height,P3^, NormalPut);
      4 : PutImage(500,Height,P4^, NormalPut);
      5 : PutImage(500,Height,P5^, NormalPut);
      6 : PutImage(500,Height,P6^, NormalPut);
      7 : PutImage(500,Height,P7^, NormalPut);
      8 : PutImage(500,Height,P8^, NormalPut);
      9 : PutImage(500,Height,P9^, NormalPut);
     10 : PutImage(500,Height,P10^,NormalPut);
     11 : PutImage(500,Height,P11^,NormalPut);
     12 : PutImage(500,Height,P12^,NormalPut);
     13 : PutImage(500,Height,P13^,NormalPut);
     14 : PutImage(500,Height,P14^,NormalPut);
     15 : PutImage(500,Height,P15^,NormalPut);
     16 : PutImage(500,Height,P16^,NormalPut);
     17 : PutImage(500,Height,P17^,NormalPut);
     18 : PutImage(500,Height,P18^,NormalPut);
    end;
    OutTextXY(500 + Offset,Height + 130,'4');
  end;

PROCEDURE SCORE_HIT;
  var
    hitstr : string;
  begin
    inc(Hits);
    SetTextStyle(defaultfont,horizdir,1);
    str(Hits,hitstr);
    hitstr := 'Hits: ' + hitstr;
    OutTextXY(550,315,hitstr);
  end;

PROCEDURE GET_CHOICE;
  begin
```

```
    repeat
      key := readkey
    until
      (key = #27) or
      (key = #49) or
      (key = #50) or
      (key = #51) or
      (key = #52);
    if (NumbX = 1) and (key = #49) then SCORE_HIT;
    if (NumbX = 2) and (key = #50) then SCORE_HIT;
    if (NumbX = 3) and (key = #51) then SCORE_HIT;
    if (NumbX = 4) and (key = #52) then SCORE_HIT;
    if key <> #27 then
      begin
        case NumbX of
          1 : begin
                Numb2 := 0;Numb3 := 0;Numb4 := 0;
              end;
          2 : begin
                Numb1 := 0;Numb3 := 0;Numb4 := 0;
              end;
          3 : begin
                Numb1 := 0;Numb2 := 0;Numb4 := 0;
              end;
          4 : begin
                Numb1 := 0;Numb2 := 0;Numb3 := 0;
              end;
        end;
        Put_Parts;
        Delay(500);
        inc(Trial);
      end
    else
      begin
        NumbS := '0';
        Send_Info;
      end;
  end;

PROCEDURE SHOW_SCORE;
  var
    key : char;
  begin
    dec(Trial);if Trial = 0 then Trial := 1;
    score := (Hits/Trial) * 100;
    rate  := 25 - score;
    if rate < 0 then rate := rate * (-1);
    if score < 25 then
```

```
        dir := 'below';
    if score > 25 then dir := 'above';
    if score = 25 then dir := 'at';
    clrscr;
    gotoxy(10,10);write('YOUR SCORE:');
    gotoxy(20,11);write('Number of trials    : ',Trial);
    gotoxy(20,12);write('Number of hits      : ',Hits);
    gotoxy(20,13);write('Chance hits         : ',Trial div 4);
    gotoxy(20,13);write('Percentage success : ',score:5:2,'%');
    gotoxy(20,14);write('Chance probability : 25%');
    gotoxy(20,16);write('You are ');
      if dir <> 'at' then write (rate:5:2,'% ',dir)
      else write(dir);
      write(' probability.');
  end;

PROCEDURE RESET_SCORE;
  begin
    NumbS := 'X';
    Trial := 1;
    Hits  := 0;
  end;

PROCEDURE TITLE;
  begin
    clrscr;
    write  ('****************************************');
    writeln('****************************************');
    write  ('*                                   MENT');
    writeln('COMM                                   *');
    write  ('*                                       ');
    writeln('                         SGFOIA3       *');
SG1J    write  ('*      [redacted] [redacted]');
    writeln(', [redacted]                           *');   SGFOIA3
    write  ('****************************************');
    writeln('****************************************');
    writeln;
  end;

PROCEDURE INSTRUCT;
  var
    key : char;
  begin
    TITLE;
    write  ('    MENTCOMM lets you practice your abil');
    writeln('ity to communicate mentally.');
    writeln;
    writeln('    MENTCOMM can be used in 2 ways:');
  writeln;
  write  ('1).  Used on a single machine, you attem');
  writeln('pt to establish mental contact with ');
  write  ('     the computer, and select the pictur');
  writeln('e it has chosen as the "TARGET".');
  writeln;
  write  ('2).  If you connect this machine, via mo');
  writeln('dem or null modem, to another computer,');
  write  ('     and run the program MENTCOM2 on it,');
  writeln(' the target picture will show up on ');
  write  ('     the other screen, and a second pers');
  writeln('on can act as a "sender" of the');
  writeln('     information.');
  writeln;
  writeln('                              Press any key to continue');
key := readkey;
  TITLE;
  write  ('   After you read this screen, you wil');
  writeln('l be asked to press a key again.  When');
  write  ('do, you will first see a library of pi');
  writeln('ctures from which the computer will');
  write  ('select during each turn of the communi');
  writeln('cation process.');
  writeln;
  write  ('   The computer will then select four ');
  writeln('pictures, and show them to you.  It');
  write  ('will also secretly select one of them ');
  writeln('as the "TARGET" photo, and wait for');
  write  ('you to get a mental "feeling" (either ');
  writeln('from the machine, or from the person');
  write  ('acting as the SENDER, as to which pict');
  writeln('ure is the TARGET.  FIRST IMPRESSIONS ');
  write  ('ARE OFTEN THE BEST.  When you get a fe');
  writeln('eling for the target, press the number');
  write  ('key corresponding to it.  If you are r');
  writeln('ight, the HITS counter will add one.');
  write  ('After your choice, the correct picture ');
  writeln('will show briefly, and then you will');
  writeln('go on to the next round.');
  writeln;
  write  ('   When you finish, press ESCAPE, and y');
  writeln('ou will be given your score.');
  writeln;
  writeln('                              Press any key to continue');
key := readkey;
  TITLE;
  write  ('   There are presently 3 libraries of ');
  writeln('"target pictures" from which to choose.');
```

```
     writeln('They are:');
     writeln;
     writeln('    1.  Miscellaneous Targets');
     writeln('    2.  Facial expressions');
     writeln('    3.  Geometric symbols');
     writeln;
     write  ('With which library would you like to work?  ');
     repeat
       key := readkey;
     until (key = '1') or
           (key = '2') or
           (key = '3');


     picfile   := 'SET1.PI3';
     picfile[4]:= key;
     write(key);
     writeln;
     writeln;
     write  ('                              PRESS ANY KEY TO BEGIN');
key := readkey;
     WRITE_LINE(PicFile);
     clrscr;
     gotoxy(29,10);
     write ('NOW SETTING THINGS UP');
     gotoxy(30,11);
     write ('ONE MOMENT, PLEASE');
     delay (2000);
   end;

 FUNCTION FILE_EXISTS(filename: string):boolean;
   var   f  : file;
     begin
       {$I-}
       assign(f,filename);
       reset(f);
       close(f);
       {$I+}
       File_Exists := (IOresult = 0) and (filename <> '');
     end;

 PROCEDURE RECORD_SCORE;
   var
     date,
     runs,
     good,
     sender,
     sess,
  library: string;
  a,z    : integer;
  f      : text;
begin
  sess   := 'Reading: ';
  sender := 'Computer';
  date   := '';
  name   := '';
  library:= copy(PicFile,1,4);
  clrscr;
  gotoxy(15,10);
  write ('PLEASE GIVE ME YOUR NAME: ');
  readln(Name);
  gotoxy(15,12);
  write ('Please give today''s date (YYMMDD):  ');
  readln(date);
  gotoxy(15,14);
  write ('Was this a solo run?  ');
  sess := 'Reading: ';
  repeat
    key := upcase(readkey);
  until (key = 'Y') or
        (key = 'N');
  if key = 'N' then
    begin
      gotoxy(15,16);
      write ('Please give sender''s name:  ');
      readln(sender);
    end;
  str(Trial,runs);
  str(Hits ,good);
  for a := 1 to length(name) do
    if name[a] = #32 then
      begin
        for z := a to length(name)-1 do
          name[z] := name[z+1];
        name := copy(name,1,length(name)-1);
      end;
  name := copy(name,1,8) + '.MNT';
  assign (f, name);
  if File_exists(name) = false then
    begin
      rewrite(f);
      close  (f);
    end;
  append(f);
  writeln(f,date,'   ',trial,'   ',hits,'   ',library,'   ',sess,sender);
  close(f);
  end;

BEGIN
  assign(Lineout,'COM2');
  INSTRUCT;
  RESET_SCORE;
  randomize;
  DECODE_PI3_PICTURE(PicFile,600,300);
  BREAK_PICTURE_UP;
  repeat
    key   := 'C';
    Graphics_Done;
    Graphics_Init;
    repeat
      PICK_4;
      PICK_1;
      SEND_INFO;
      PUT_PARTS;
      GET_CHOICE;
    until key = #27;
    Graphics_Done;
    SHOW_SCORE;
    gotoxy(20,19);write('Press Q to end the program.');
    gotoxy(20,20);write('Press A to start again.');
    gotoxy(20,21);write('Press C to continue.');
    repeat
      key := readkey;
      key := upcase(key);
    until
      (key = 'Q') or
      (key = 'A') or
      (key = 'C');
    Action := key;
    WRITE_LINE(Action);
    if key = 'C' then NumbS := '1';
    if key = 'A' then RESET_SCORE;
  until key = 'Q';
  clrscr;
  RECORD_SCORE;
  clrscr;
END.
```

SOURCE CODE FOR THE PROGRAM MENTCOM2.EXE
WRITTEN IN TURBO PASCAL Ver. 5.1

```
PROGRAM MENTCOMM;
  {$I-}
  USES
    CRT, DOS, GRAPH, PI3DECOD, NUMBCONV;

  CONST
    Height = 100;
    Width  = 100;
    Offset =  50;

  VAR
    code     : integer;
    Dir      : string;
    dummy    : string;
    Key      : string;
    Hits     : integer;
    Lineout  : text;
    NumbX    : integer;
    NumbH,
    NumbS,
    NumbT    : string;
    P1, P2,
    P3, P4,
    P5, P6,
    P7, P8,
    P9, P10,
    P11,P12,
    P13,P14,
    P15,P16,
    P17,P18  : pointer;
    PicFile  : string;
    Rate     : real;
    Score    : real;
    Size     : word;
    Trial    : integer;
    X1, X2,
    X3, X4,
    X5, X6,
    Y1, Y2,
    Y3       : integer;

  FUNCTION  SETX(Number : integer) : integer;
    begin
      case Number of
```

```
       1,  7, 13 : SETX :=   0;
       2,  8, 14 : SETX := Width;
       3,  9, 15 : SETX := Width * 2;
       4, 10, 16 : SETX := Width * 3;
       5, 11, 17 : SETX := Width * 4;
       6, 12, 18 : SETX := Width * 5;
    end;
  end;

FUNCTION SETY(Number : integer) : integer;
  begin
    case Number of
       1,  2,  3,  4,  5,  6 : SETY :=   0;
       7,  8,  9, 10, 11, 12 : SETY := Height;
      13, 14, 15, 16, 17, 18 : SETY := Height * 2;
    end;
  end;

FUNCTION READ_LINE:string;
  var
    tempstr : string;
  begin
    reset (Lineout);
    readln(Lineout,tempstr);
    if tempstr[1] = #10 then
      tempstr := copy(tempstr,2,length(tempstr)-1);
    close (Lineout);
    READ_LINE := tempstr;
  end;

PROCEDURE GET_INFO;
  begin
    NumbS := 'X';NumbT := 'X';NumbH := 'X';
    repeat NumbS := READ_LINE until ioresult = 0;
    Val(NumbS,NumbX,code);
    repeat NumbT := READ_LINE until ioresult = 0;
    Val(NumbT,Trial,code);
    repeat NumbH := READ_LINE until ioresult = 0;
    Val(NumbH,Hits ,code);
  end;

PROCEDURE BREAK_PICTURE_UP;
  var
    x : integer;
    y : integer;
  begin
    Size := ImageSize(00,00,Width,Height);
    GetMem(P1, Size); X := SETX( 1); Y := SETY( 1);
    GetImage(X, Y, X + Width, Y + Height, P1^);
    GetMem(P2, Size); X := SETX( 2); Y := SETY( 2);
    GetImage(X, Y, X + Width, Y + Height, P2^);
    GetMem(P3, Size); X := SETX( 3); Y := SETY( 3);
    GetImage(X, Y, X + Width, Y + Height, P3^);
    GetMem(P4, Size); X := SETX( 4); Y := SETY( 4);
    GetImage(X, Y, X + Width, Y + Height, P4^);
    GetMem(P5, Size); X := SETX( 5); Y := SETY( 5);
    GetImage(X, Y, X + Width, Y + Height, P5^);
    GetMem(P6, Size); X := SETX( 6); Y := SETY( 6);
    GetImage(X, Y, X + Width, Y + Height, P6^);
    GetMem(P7, Size); X := SETX( 7); Y := SETY( 7);
    GetImage(X, Y, X + Width, Y + Height, P7^);
    GetMem(P8, Size); X := SETX( 8); Y := SETY( 8);
    GetImage(X, Y, X + Width, Y + Height, P8^);
    GetMem(P9, Size); X := SETX( 9); Y := SETY( 9);
    GetImage(X, Y, X + Width, Y + Height, P9^);
    GetMem(P10,Size); X := SETX(10); Y := SETY(10);
    GetImage(X, Y, X + Width, Y + Height, P10^);
    GetMem(P11,Size); X := SETX(11); Y := SETY(11);
    GetImage(X, Y, X + Width, Y + Height, P11^);
    GetMem(P12,Size); X := SETX(12); Y := SETY(12);
    GetImage(X, Y, X + Width, Y + Height, P12^);
    GetMem(P13,Size); X := SETX(13); Y := SETY(13);
    GetImage(X, Y, X + Width, Y + Height, P13^);
    GetMem(P14,Size); X := SETX(14); Y := SETY(14);
    GetImage(X, Y, X + Width, Y + Height, P14^);
    GetMem(P15,Size); X := SETX(15); Y := SETY(15);
    GetImage(X, Y, X + Width, Y + Height, P15^);
    GetMem(P16,Size); X := SETX(16); Y := SETY(16);
    GetImage(X, Y, X + Width, Y + Height, P16^);
    GetMem(P17,Size); X := SETX(17); Y := SETY(17);
    GetImage(X, Y, X + Width, Y + Height, P17^);
    GetMem(P18,Size); X := SETX(18); Y := SETY(18);
    GetImage(X, Y, X + Width, Y + Height, P18^);
  end;

PROCEDURE PUT_TARGET;
  var
    trialstr : string;
    hitstr   : string;
  begin
    ClearDevice;
    str(Trial,trialstr);
    trialstr := 'Trial # ' + trialstr;
    str(Hits ,hitstr);
    hitstr   := 'Hits: ' + hitstr;
    SetTextStyle(defaultfont,horizdir,1);
```

```
   {ACTIVATE NEXT 2 LINES TO SHOW BOTH TRIALS AND HITS}
   {OutTextXY(58,315,trialstr);
    OutTextXY(550,315,hitstr);}
    SetTextStyle(gothicfont,horizdir,3);
   {ACTIVATE NEXT LINE TO SHOW ONLY HITS}
    OutTextXY(300,315,hitstr);
    case NumbX of
      1 : PutImage(275,Height,P1^, NormalPut);
      2 : PutImage(275,Height,P2^, NormalPut);
      3 : PutImage(275,Height,P3^, NormalPut);
      4 : PutImage(275,Height,P4^, NormalPut);
      5 : PutImage(275,Height,P5^, NormalPut);
      6 : PutImage(275,Height,P6^, NormalPut);
      7 : PutImage(275,Height,P7^, NormalPut);
      8 : PutImage(275,Height,P8^, NormalPut);
      9 : PutImage(275,Height,P9^, NormalPut);
     10 : PutImage(275,Height,P10^,NormalPut);
     11 : PutImage(275,Height,P11^,NormalPut);
     12 : PutImage(275,Height,P12^,NormalPut);
     13 : PutImage(275,Height,P13^,NormalPut);
     14 : PutImage(275,Height,P14^,NormalPut);
     15 : PutImage(275,Height,P15^,NormalPut);
     16 : PutImage(275,Height,P16^,NormalPut);
     17 : PutImage(275,Height,P17^,NormalPut);
     18 : PutImage(275,Height,P18^,NormalPut);
    end;
  end;

PROCEDURE SHOW_SCORE;
  var
    key : char;
  begin
    dec(Trial);if Trial = 0 then Trial := 1;
    score := (Hits/Trial) * 100;
    rate  := 25 - score;
    if rate < 0 then rate := rate * (-1);
    if score < 25 then
        dir := 'below';
    if score > 25 then dir := 'above';
    if score = 25 then dir := 'at';
    clrscr;
    gotoxy(10,10);write('SCORE:');
    gotoxy(20,11);write('Number of trials   : ',Trial);
    gotoxy(20,12);write('Number of hits     : ',Hits);
    gotoxy(20,13);write('Chance hits        : ',Trial div 4);
    gotoxy(20,13);write('Percentage success : ',score:5:2,'%');
    gotoxy(20,14);write('Chance probability : 25%');
    gotoxy(20,16);write('You are ');
```

```
      if dir <> 'at' then write (rate:5:2,'% ',dir)
      else write(dir);
      write(' probability.');
  end;

PROCEDURE INSTRUCT;
  begin
    clrscr;
    gotoxy (36, 3);
    write  ('MENTCOM2');
    gotoxy ( 1, 5);
    write  ('   This portion of the MENTCOMM progra');
    writeln('m allows you to practice and develop');
    write  ('your ability to send mental messages to');
    writeln(' another person.  MENTCOM2 works only');
    writeln('in conjunction with the MENTCOMM program.');
    writeln;
    write  ('   When the "receiving" person has read');
    writeln(' their instructions, you will be shown');
    writeln('a screen of possible target pictures.');
    writeln;
    write  ('   The "receiving" person will see a pa');
    writeln('nel of 4 of these pictures, chosen at ');
    write  ('random, one of which will be the "TARGE');
    writeln('T" picture (they will not know which).');
    write  ('Only you will see the "TARGET" picture.');
    writeln('  You should concentrate on it to the');
    write  ('best of your ability.  You do not nece');
    writeln('ssarily have to "SEND" it to the other');
    write  ('person.  It is often best to just expe');
    writeln('rience it as fully as you can, losing');
    write  ('yourself in that experience, and let ');
    writeln('the receiving person do the rest.');
    writeln;
    write  ('The program will proceed as soon as the');
    writeln(' receiving person is ready.');
    writeln;
    repeat PicFile := READ_LINE until ioresult = 0;
    clrscr;
  end;

BEGIN
  assign(Lineout,'COM2');
  INSTRUCT;
  DECODE_PI3_PICTURE(PicFile,600,300);
  BREAK_PICTURE_UP;
  repeat
    key   := 'C';
```

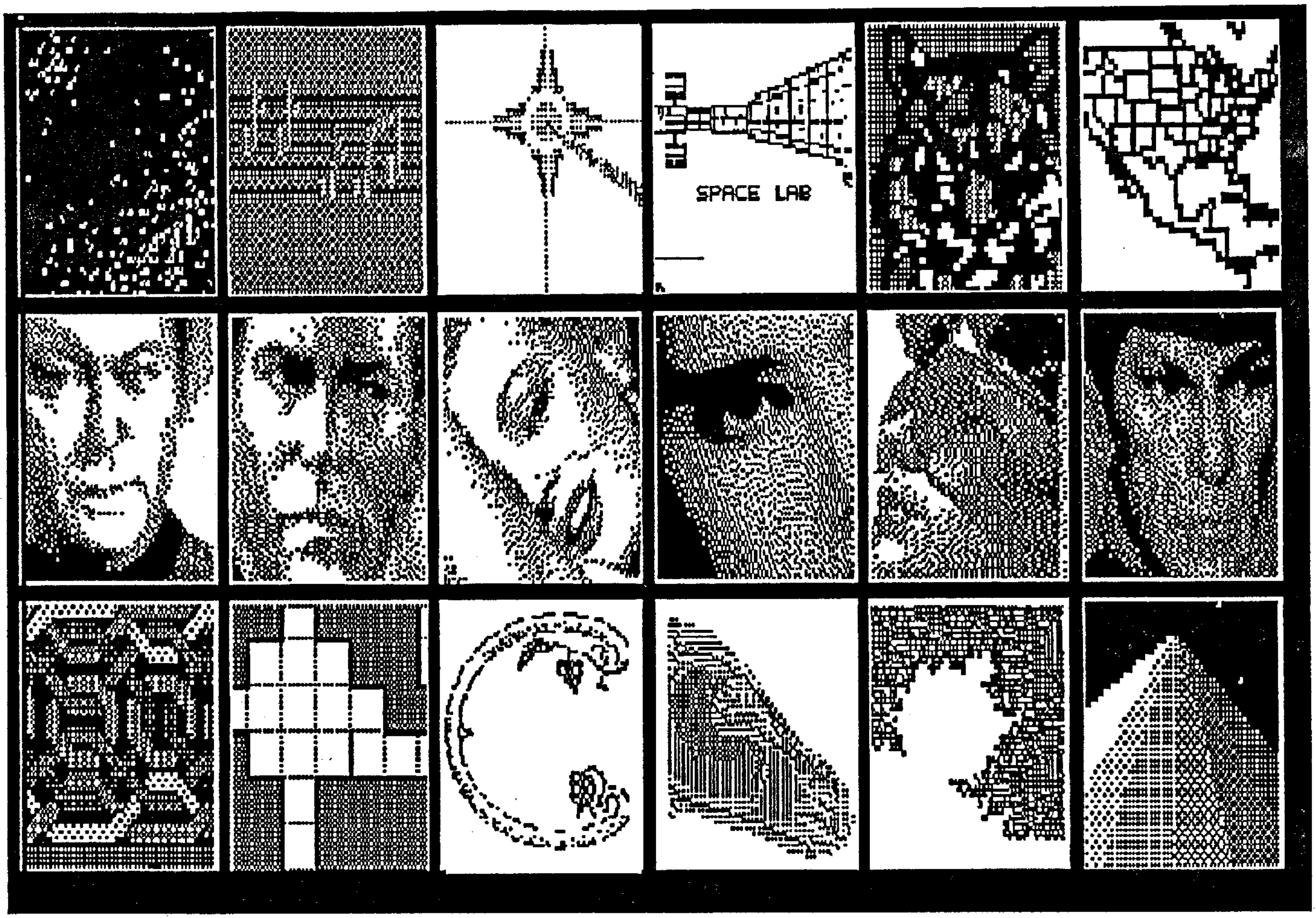
```
      Graphics_Done;
      Graphics_Init;
      repeat
        GET_INFO;
        if NumbS <> '0' then PUT_TARGET;
      until NumbS = '0';
      Graphics_Done;
      SHOW_SCORE;
      repeat key := READ_LINE until ioresult = 0;
    until key = 'Q';
    clrscr;
END.
```

SOURCE CODE FOR THE PROGRAM PI3DECOD.TPU
WRITTEN IN TURBO PASCAL Ver. 5.1

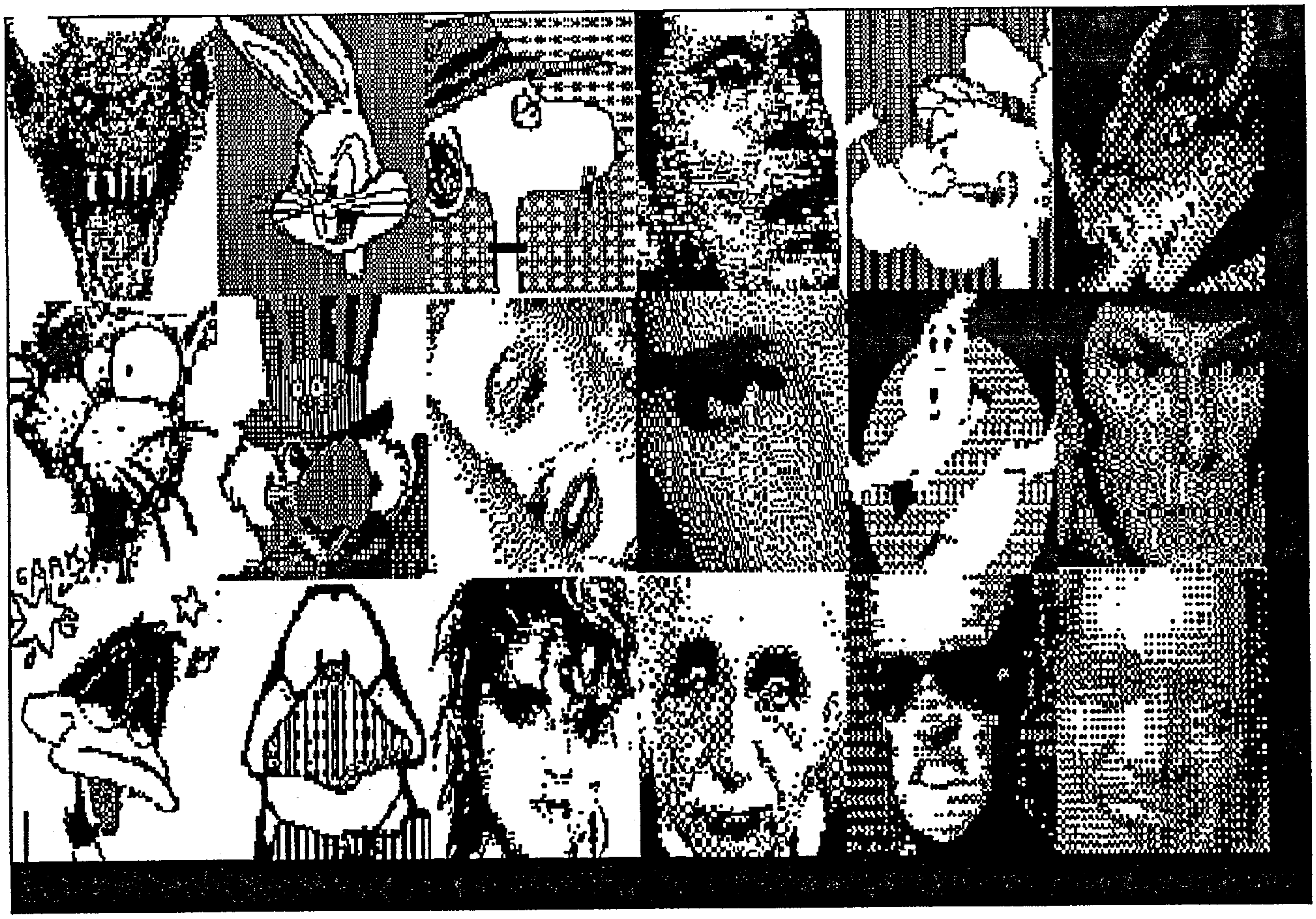
```
UNIT PI3DECOD;

  INTERFACE

  USES
    CRT, GRAPH;

    PROCEDURE GRAPHICS_INIT;
    PROCEDURE GRAPHICS_DONE;
    PROCEDURE DECODE_PI3_PICTURE(FileName:string;Pixwide,Pixhigh:integer);

  IMPLEMENTATION
  VAR
    F            : text;
    FileName     : string;

    Procedure Graphics_Init;
      var
        GraphDriver : integer;
        GraphMode   : integer;
        ErrorCode   : integer;
      begin
        GraphDriver := Detect;
        InitGraph(GraphDriver, GraphMode,'');
        ErrorCode := GraphResult;
        if ErrorCode <> grok then
          begin
            writeln('ERROR: ',GraphErrorMsg(ErrorCode));
            writeln('Either that file does not exist, or there is a');
            writeln('graphics error.  You may not have a graphics ');
            writeln('card, or the proper .BGI file.');
            Writeln('Program aborted....');
            Readln;
            Halt(1);
          end;
      end;  {Sub Procedure Graphics_Init}

    Procedure Graphics_Done;
      begin
        CloseGraph
      end; {Sub Procedure Graphics_Done}

   PROCEDURE DECODE_PI3_PICTURE(FileName:string;Pixwide,Pixhigh:integer);
   var
```

```
    B            : char;
    color1       : integer;
    color2       : integer;
    TotalBytes   : integer;
    X            : integer;
    XSize        : integer;
    Y            : integer;
    YSize        : integer;
    BitStr : string[8];
    Count  : integer;
    Times  : integer;
    Total  : integer;
    tempint: integer;

PROCEDURE OPEN_FILE;
  begin
    assign(f,filename);
    {$I-}
    reset(f);
    {$I+}
  end;

  Procedure Set_Parameters;
    begin
      XSize := pixwide;
      YSize := pixhigh;
      TotalBytes := XSize div 8;
    end;  {Sub Procedure Set_Parameters}

  Procedure Display_Byte;
    var
      bytes : integer;
    begin
      for bytes := 1 to 8 do
        begin
          if BitStr[bytes] = '1' then putpixel(x,y,0)
          else putpixel(x,y,1);
          inc(x);
        end;
    end;  {Sub Procedure Display_Byte}

  Function  Byte_To_Bits(ByteOrd: integer):string;
    var
      place   : integer;
      amt     : integer;
      tempstr : string[8];
    begin
      amt    := 128;
    place := 1;
    for place := 1 to 8 do
      begin
        if ByteOrd - amt < 0 then
          tempstr[place] := '0'
        else
          begin
            tempstr[place] := '1';
            ByteOrd := ByteOrd - amt;
          end;
        amt := amt div 2;
      end;
  BYTE_TO_BITS := tempstr;
end; {Sub Function Byte_To_Bits}

Function Bits_To_Byte(instr:string):integer;
  var
    bit  : integer;
    value: integer;
    temp : integer;
  begin
    temp := 0;
    value  := 128;
    for bit  := 1 to 8 do
      begin
        if instr[bit] = '1' then temp := temp + value;
        value := value div 2;
      end;
    Bits_To_Byte := temp;
  end; {Sub Function Bits_To_Byte}

Procedure Get_File_Info;
  var filex : integer;
  begin
    for filex := 1 to 34 do
      begin
        read(f,b);
        case filex of
          3 : tempint := ord(b);
          4 : color1 := (256 * tempint) + ord(b);
          5 : tempint := ord(b);
          6 : color2 := (256 * tempint) + ord(b);
        end;
      end;
    end;

begin  {Procedure Decode_Picture}
  Open_File;
```

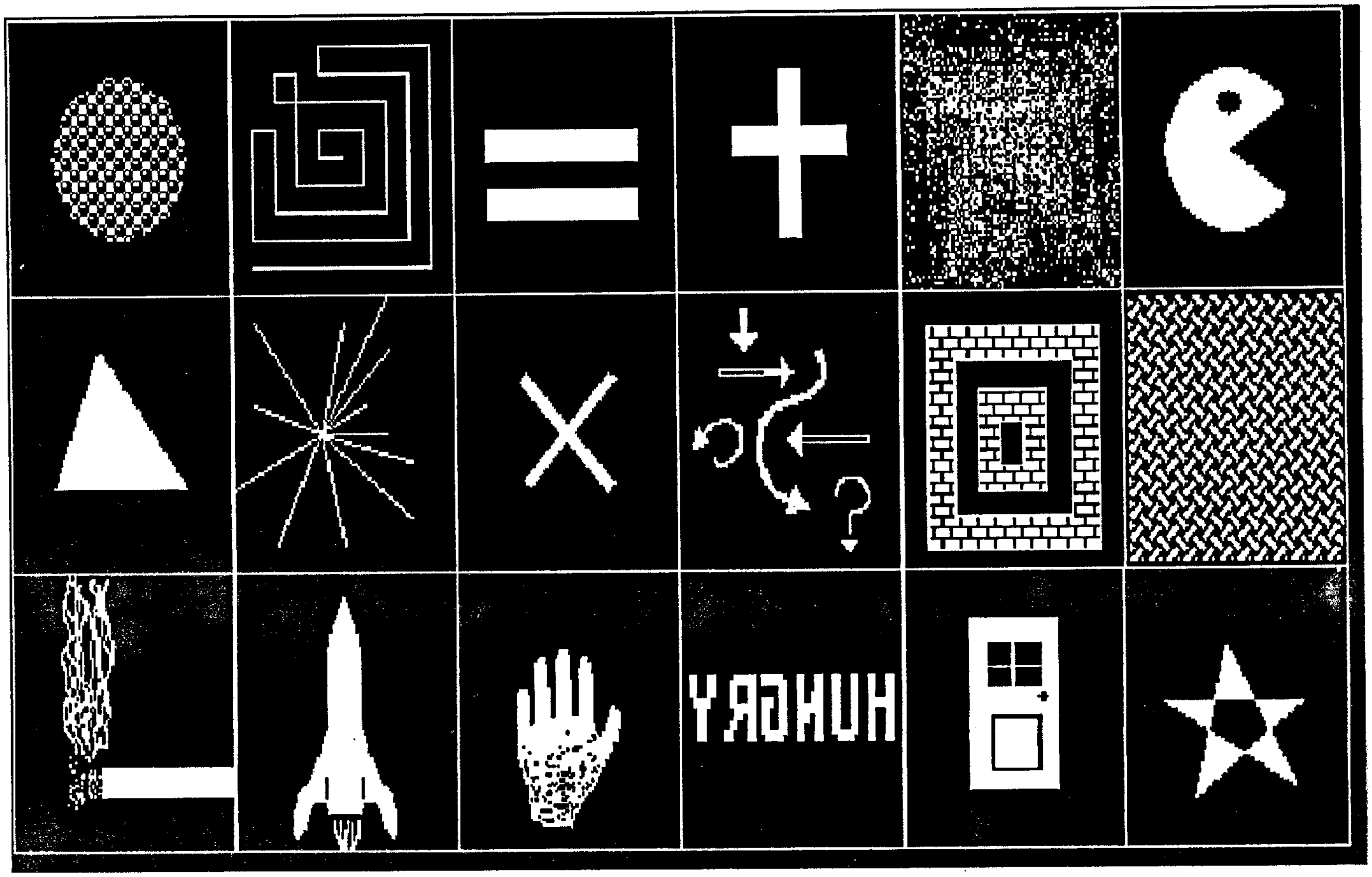
```
    Graphics_Init;
    Set_Parameters;
    for x := 1 to 34 do read(f,b);
    for y := 1 to YSize do
      begin
        x := 1;
        if keypressed then
          begin
            Graphics_Done;
            exit;
          end;
        for count := 1 to TotalBytes  do
          begin
            read(f,b);
            BitStr := Byte_To_Bits(ord(b));
            Display_Byte;
          end;
        if TotalBytes < 80 then
          for count := TotalBytes to 79 do
            read(f,b);
      end;
  end; {Procedure Decode_PI3_Picture}

end.  {UNIT PI3DECOD}
```



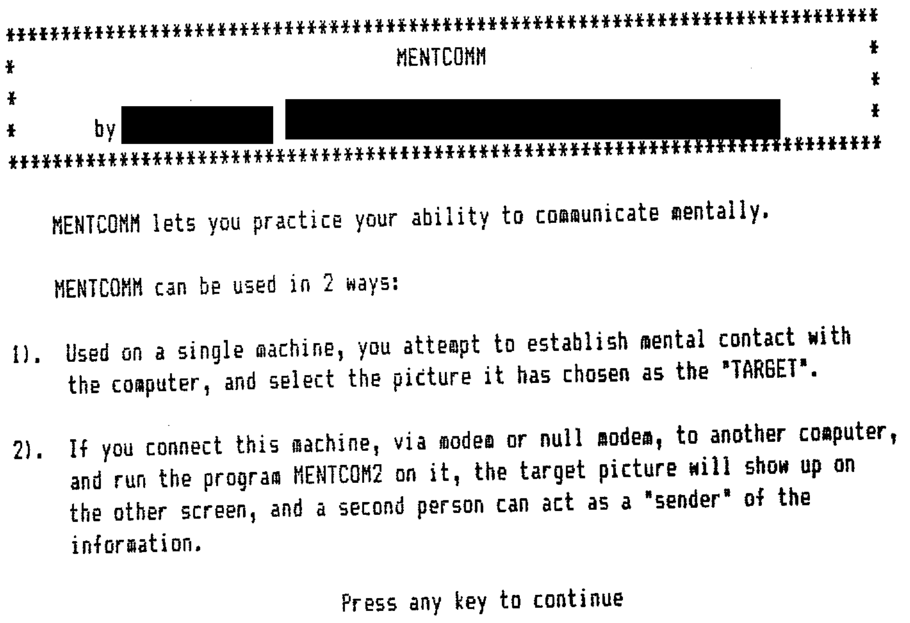
LIBRARY 1 : Miscellaneous subjects



LIBRARY 2 : Facial expressions



LIBRARY 3 : SYMBOLS



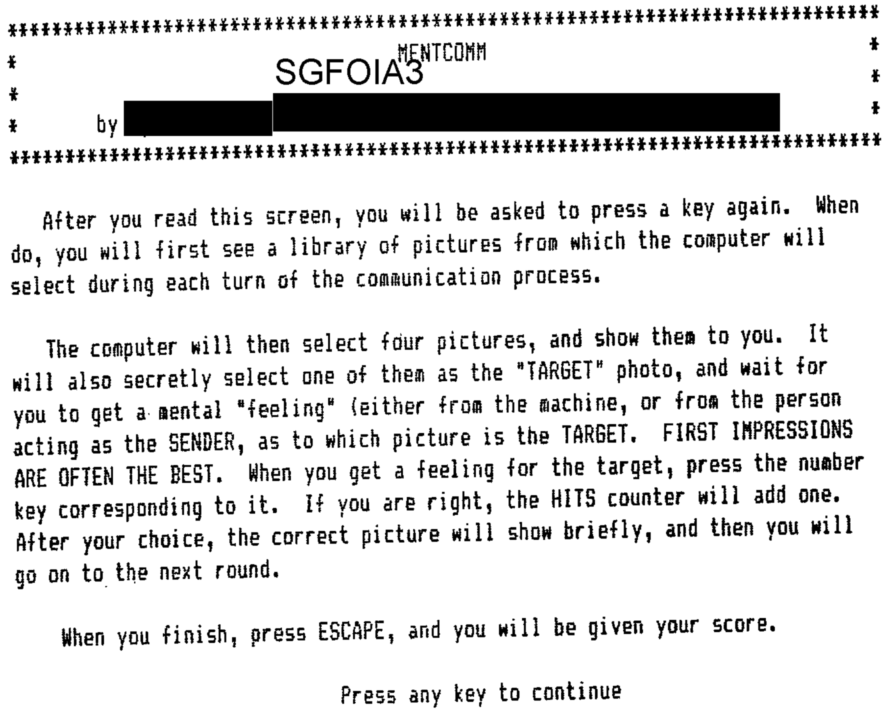
SG1J

```
****************************************************************************
*                              MENTCOMM                                    *
*                                                                          *
*                                                                          *
*      by [redacted] [redacted]
****************************************************************************
```

SGFOIA3

```
  MENTCOMM lets you practice your ability to communicate mentally.

  MENTCOMM can be used in 2 ways:

1).  Used on a single machine, you attempt to establish mental contact with
     the computer, and select the picture it has chosen as the "TARGET".

2).  If you connect this machine, via modem or null modem, to another computer,
     and run the program MENTCOM2 on it, the target picture will show up on
     the other screen, and a second person can act as a "sender" of the
     information.

                          Press any key to continue
```

SCREEN 1 : Introduction



SGFOIA3

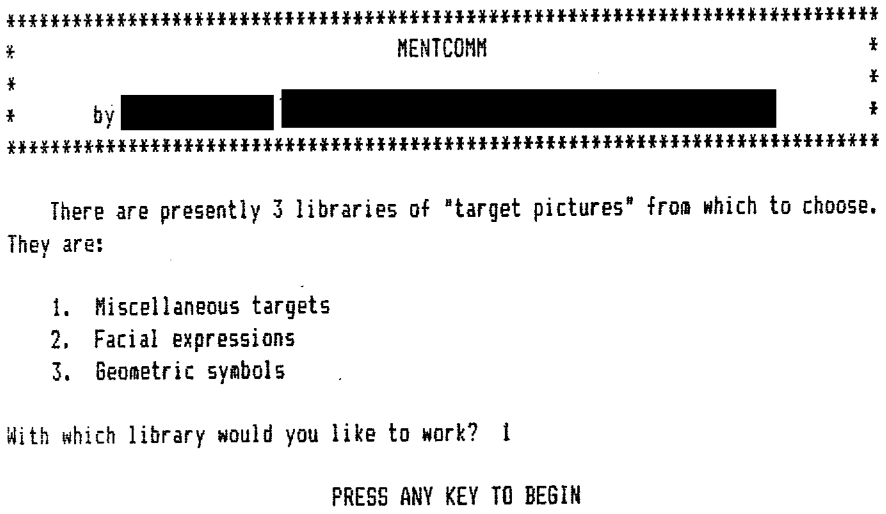
SG1J

```
***************************************************************************
*                                MENTCOMM                                 *
*                                                                         *
*       by [redacted]                                                     *
***************************************************************************

   After you read this screen, you will be asked to press a key again.  When
do, you will first see a library of pictures from which the computer will
select during each turn of the communication process.

   The computer will then select four pictures, and show them to you.  It
will also secretly select one of them as the "TARGET" photo, and wait for
you to get a mental "feeling" (either from the machine, or from the person
acting as the SENDER, as to which picture is the TARGET.  FIRST IMPRESSIONS
ARE OFTEN THE BEST.  When you get a feeling for the target, press the number
key corresponding to it.  If you are right, the HITS counter will add one.
After your choice, the correct picture will show briefly, and then you will
go on to the next round.

     When you finish, press ESCAPE, and you will be given your score.

                           Press any key to continue
```

SCREEN 2 : Instructions

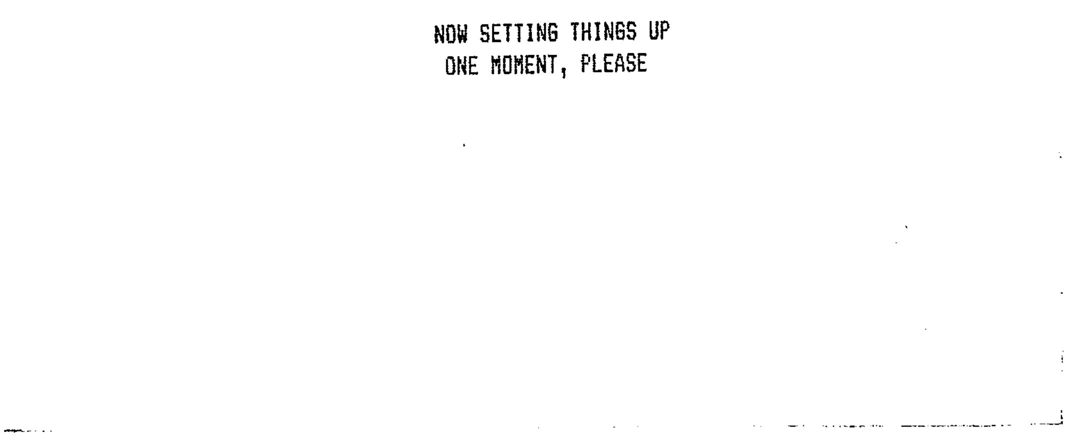
SG1J

SGFOIA3



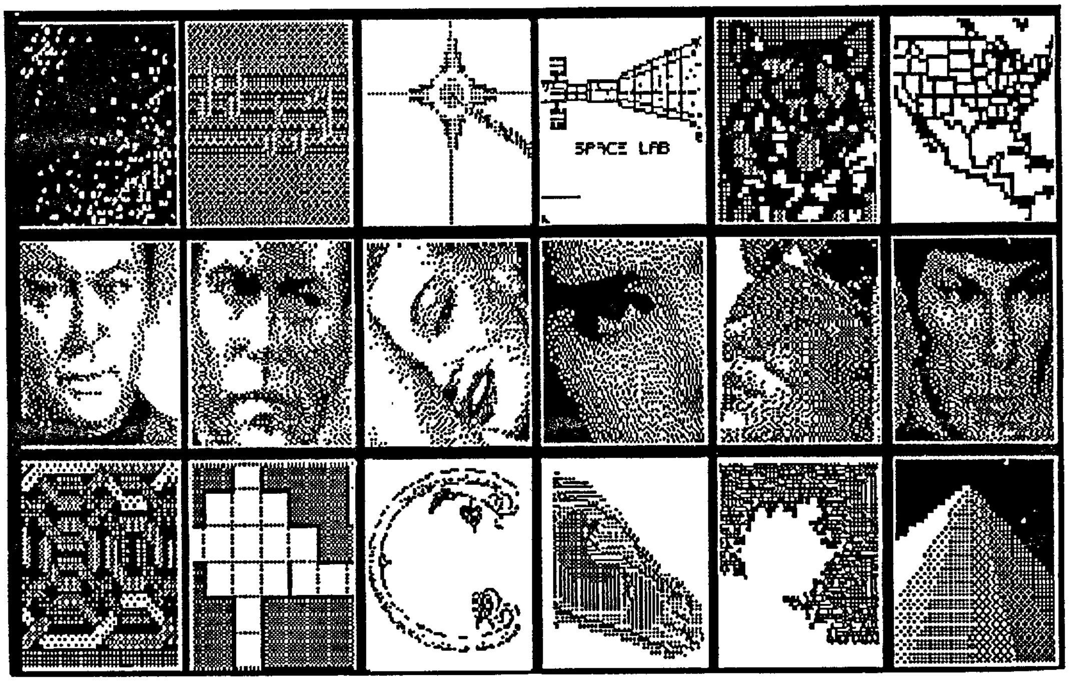
```
****************************************************************************
*                                   MENTCOMM                               *
*                                                                          *
*      by [redacted] [redacted]                                            *
****************************************************************************

    There are presently 3 libraries of "target pictures" from which to choose.
They are:

    1.  Miscellaneous targets
    2.  Facial expressions
    3.  Geometric symbols

With which library would you like to work?  1

                             PRESS ANY KEY TO BEGIN
```

SCREEN 3 : Choose Library

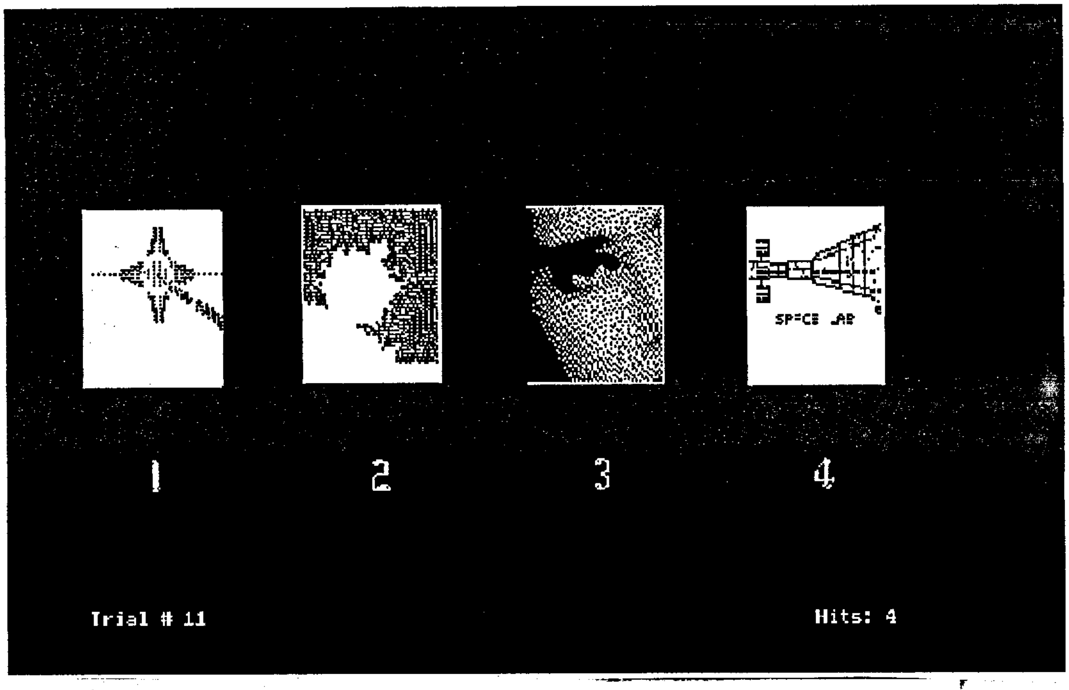


NOW SETTING THINGS UP
ONE MOMENT, PLEASE

SCREEN 4 : Notice to set up timing



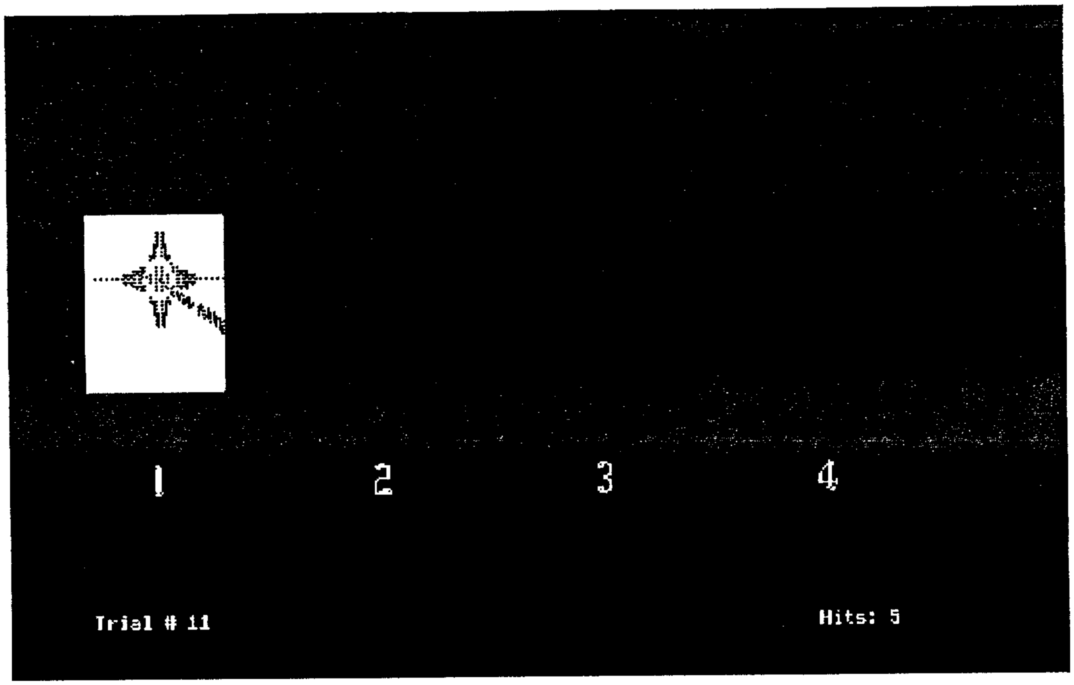
SCREEN 5 : Display of entire library



SCREEN 6 : DETECTION OF A TARGET CARD

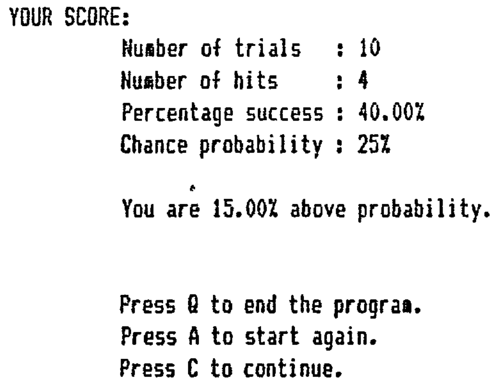
The computer randomly chooses four "cards" from the library and displays them on the screen. One is then randomly selected to be the "target". It is the task of the "receiver" participant to mentally determine which of the four is the desired target card.

This screen is taken from an actual session. One of these cards is the target. Can you determine which? (answer next page)



SCREEN 7 : IMMEDIATE FEEDBACK IS GIVEN

The target card from the previous screen was #1.
As you can see from the Hits number in the
lower right corner of the screen, the
participant correctly detected the target.
Did you?

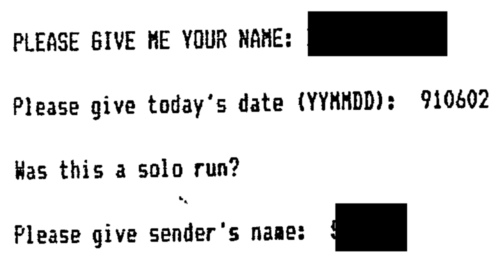


```
YOUR SCORE:
          Number of trials   : 10
          Number of hits     : 4
          Percentage success : 40.00%
          Chance probability : 25%

          You are 15.00% above probability.


          Press Q to end the program.
          Press A to start again.
          Press C to continue.
```

SCREEN 7 : Evaluation of session

SG1J

SG1J



```
PLEASE GIVE ME YOUR NAME: [redacted]

Please give today's date (YYMMDD):  910602

Was this a solo run?

Please give sender's name: [redacted]
```

SCREEN 8 : Information needed for statistics

FILENAME: [redacted]MNT

SG1J

DATE	TRIALS	HITS	LIBRARY	SENDER
910602	20	9	SET3	Smith
910603	40	17	SET1	Computer
910604	20	5	SET3	Computer
910605	100	41	SET2	Computer
910610	40	23	SET1	Smith
910613	20	7	SET1	Computer

FORMAT FOR THE PERMANENT DATA FILE